Speech and Gesture Understanding in a Homeostatic Control Framework for a Robotic Chandelier

By

Joshua Juster

Submitted to the Department of Electrical Engineering and Computer Science In partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2004

Signature of Author

Department of Electrical Engineering and Computer Science
August 26, 2004

Certified by

Deb Roy
Professor
Supervisor

Accepted by

Smith
Professor of Electrical Engineering and Computer Science
Chairman, Department Committee on Graduate Students

Speech and Gesture Understanding in a Homeostatic Control Framework for a Robotic Chandelier

By

Joshua Juster

Submitted to the Department of Electrical Engineering and Computer Science In partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

ABSTRACT

We describe a home lighting robot that uses directional spotlights to create complex lighting scenes. The robot senses its visual environment using a panoramic camera and attempts to maintain its target goal state by adjusting the positions and intensities of its lights. Users can communicate desired changes in the lighting environment through speech and gesture (e.g., "Make it brighter over there"). Information obtained from these two modalities are combined to form a goal, a desired change in the lighting of the scene. This goal is then incorporated into the system's target goal state. When the target goal state and the world are out of alignment, the system formulates a sensorimotor plan that acts on the world to return the system to homeostasis.

Thesis Supervisor: Deb Roy

Title: Associate Professor

Acknowledgements

I would like to thank my advisor, Deb Roy, for his advice, support, and constant feedback. His insights and vision were invaluable, and greatly helped steer and focus the project into its current form. I would also like to thank the members of my group, Cognitive Machines, for being so supportive and providing a great deal of help along the way. Particularly, I would like to thank Peter Gorniak, Brian Whitman, Dan Ramage, and Chris Lucas for their help with several key parts of the project. I'd also like to thank Chris and my other officemate, Jeff Bartelma, for not only providing a great deal of advice and comic relief, but also managing to tolerate the constant lighting changes and conversations with Elvis throughout the year.

I would like to thank Dan Paluska for building an amazing robot, and for his constant help in maintaining it. I'd also like to acknowledge Philip Kwok and the team at Sun for their constant work on the Sphinx speech recognizer and being quick to respond to feature requests.

Finally, I'd like to thank my parents and sister, Alex, for always being there for me and providing me with a great deal of encouragement along the way.

CONTENTS

1. Introduction

Elvis is a robotic chandelier capable of creating and maintaining complex lighting environments. The system has a target goal state which it tries to preserve by constantly monitoring its environment and adjusting its motors and lights when it detects changes that are beyond its tolerable limits. The control strategy underlying Elvis is closely related to classic cybernetic systems in which closed-loop feedback is used to maintain homeostasis. Changes in the lighting environment trigger action in the robot, which tries to compensate and thus maintain its target lighting conditions. Users can affect change in the robot's target goal state through speech and gesture, causing it to go into action to regain homeostasis.

One of the key distinctions between Elvis and other robotic systems is the fact that it maintains an intermediate goal state that is directly affected by user interaction. There is no direct mapping from commands to actions. A request for a change in lighting is translated into a goal, which is then incorporated into Elvis' overall expectations of the world, referred to as the *target goal state.* The incorporation of user input into the system's target goal state is referred to as *goal shifting.* When Elvis senses a sizable difference between its target goal state and the actual world, it activates its action planning mechanism to select its optimal lighting configuration. We refer to this process as *goal maintenance.* Figure 1 illustrates the processes.

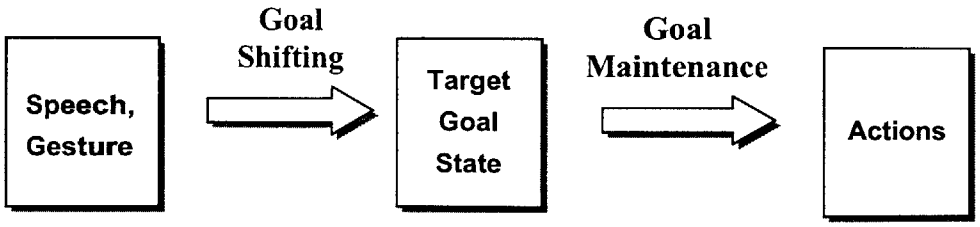


Figure 1. Mapping user input to goals

Goal shifting, itself, can be thought of as a two-step process. In the first step, the system determines how to appropriately map speech and gesture into a goal. In the second step, the system must map this goal onto its target goal state.

Goal maintenance is not only performed when a change is made to the target goal state. Instead, Elvis senses differences between its target goal state and its view of the world. As a result, a change in the system's world state will also spark the system's action planning mechanism to reconfigure its lights. If, for example, the lighting landscape is altered when the lights in the neighboring room are turned off, the system would adjust its lights to compensate for this change.

In order to maintain a homeostatic state, the system responds to differences in its target goal state and its world state by adjusting its lights. This property is characteristic of systems in first-order cybernetics. A negative feedback loop is created, in which a corrective action is taken whenever the world state deviates significantly from the target goal. The target goal, however, is not preset, and is actively changed with user input. This feedback loop can be seen in Figure 2.

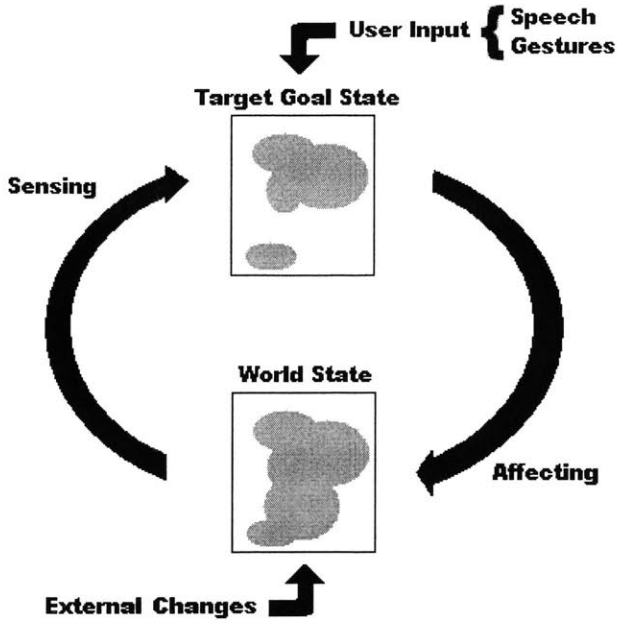


Figure 2. Negative feedback loop

The separation of goals and actions allow the system to be very flexible. The system makes decisions based on the aggregation of all prior goals. When goal shifting, there is no notion of the physical features of the system, such as how many lights are available or how they effect the environment. In other words, if a user requests light on a certain area, the system does not assign the task to a specific light. Instead, the system devises a plan of action based on its current overall state. This enables the system to be able to handle circumstances in which its hardware might not be fully adequate. For example, if the user demands that light be placed in more areas than the system is capable of covering with its spotlights, Elvis will manage to find a stable arrangement which best satisfies its requirements. If commands were to be mapped directly to actions, when the system ran out of lights, it would likely fail.

This also means that the user never has to be concerned with the system's hardware configuration. Rather than expressing which lighting element to control, the

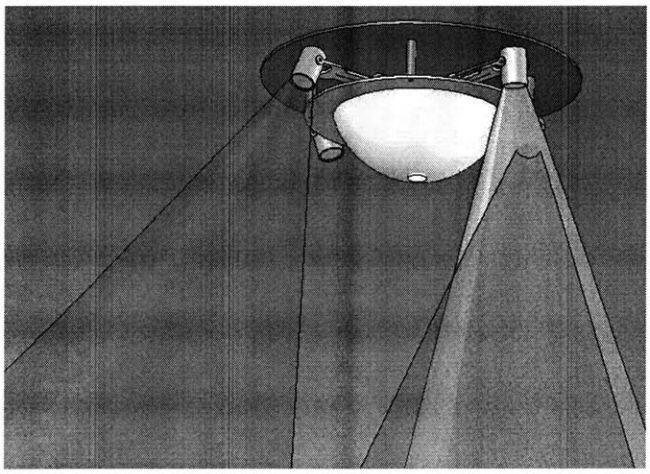
user expresses desired lighting conditions using speech and gesture, and leaves it to Elvis to map those desires into specific actions. As described in Section 3.1, Elvis learns how to map goals into actions by going through a training phase where it learns how its motors and lights effect the environment. As a result, changing the system's hardware, such as adding and removing spotlights or enabling Elvis to control electric window blinds, would involve little more than retraining the system.

Although this paper will focus on the implementation and properties of a goal maintaining system, the value of a speech and gesture interface for home lighting should not be overlooked. The featured system enables users to describe *what* they would like their environment to be like and not worry about *how* the system accomplishes this feat. Users can request any amount of light to be added or removed from anywhere in the room. Without a natural interface, such a chore would be unwieldy. In fact, a recent study showed that users would prefer using speech or a combination of speech and gesture as an interface for home lighting control compared to nearly every other type of interface, including automatic sensing, computer wall displays, touch lights, normal switches, and the clapper [6].

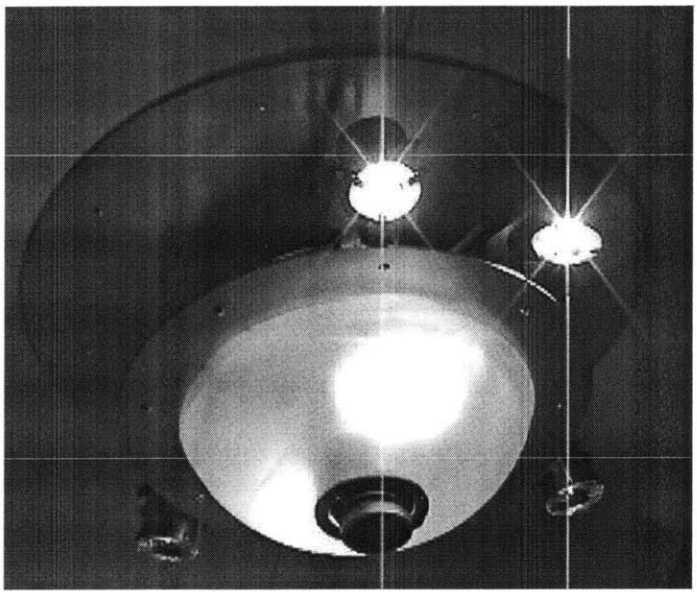
In this paper, we will first describe Elvis' physical embodiment and the process of training the system. Next, we will proceed to focus on our representation of goals. We will describe goal shifting: first, we will explain how speech and gesture are recognized and then will describe how they are mapped into goals. We will then describe goal maintenance and how the action planning mechanism transforms goals into light and motor actions. Finally, we will conclude by describing an evaluation of the current system.

2. Embodiment

Elvis' hardware consists of a custom-built robotic lighting fixture consisting of four two degree-of-freedom (DOF) directed spotlights, all of which swivel around a central ambient light. A video camera with fish eye lens is mounted in the center of the domed central light. Figure 3 shows (a) a design sketch of the lighting system with directed beams of light and (b) a photo of the actual device. Each of Elvis' spotlights is capable of reaching approximately 40% of a 25 ft. by 25 ft. room. They can tilt 90° and rotate 180° around the ambient light.



(a) Design Sketch of Elvis



(b) Photo of Elvis

Figure 3. Elvis' hardware embodiment.

Each spotlight is independently controlled by two servomotors. The first DOF allows each light to move radially along the perimeter of the central dome. Each light can pan approximately 160 degrees. This large range creates the opportunity for lights to collide with one another (see Figure 4).

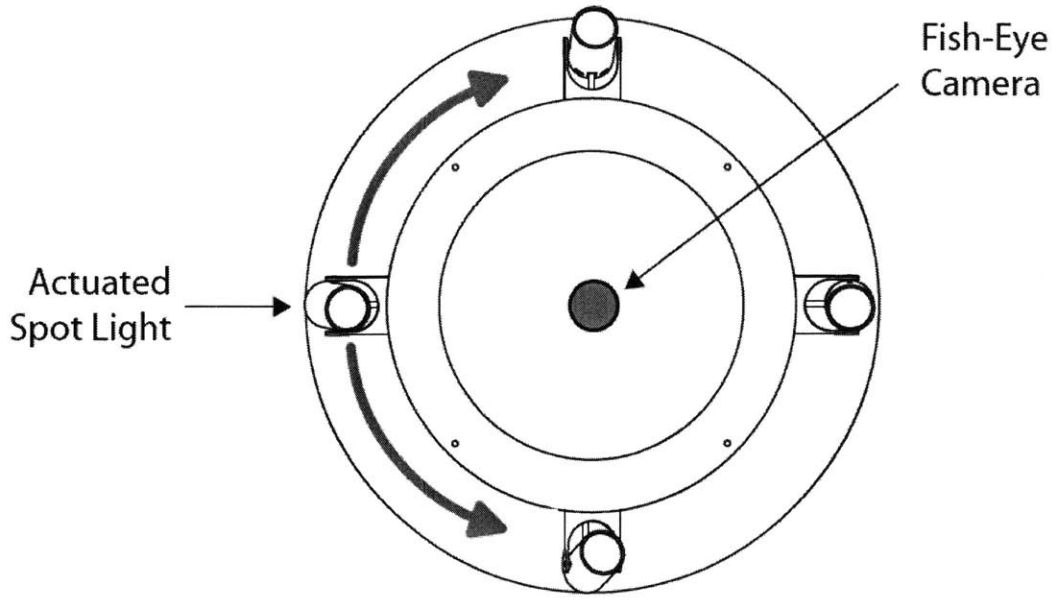


Figure 4. A bottom view of Elvis

The second DOF provides a pivot motion along the vertical plane. The light can tilt approximately 90 degrees, as shown in Figure 5.

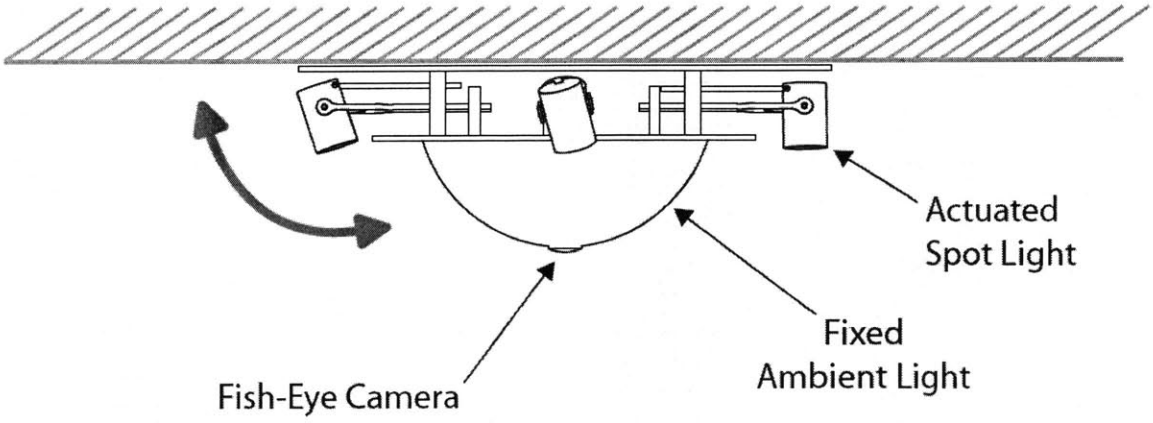


Figure 5. A side view of Elvis

The intensity of each lamp (four spots and one central ambient in total) is under computer control with a resolution of 127 intensity settings.

The spotlights are fairly precise, projecting an 8-degree beam that creates a spot with a radius of approximately 2 foot spot when mounted on an 8 foot ceiling. There is a rapid falloff of illumination outside the focal area.

The user wears a wireless microphone in order to interact with the system. Speech is converted to text using the Sphinx 4 speech recognizer [1]. The recognizer is trained on a trigram model of approximately sixty words. A full list of the vocabulary can be found in Appendix A.

3. Sensorimotor Learning

Elvis utilizes direct inverse modeling [2] in order to learn how its spotlights affect its environment. A training phase involves two stages: motor babbling and scene analysis. First, the system captures an image of the environment as viewed through its camera with all of its lights turned off. Elvis then activates one of its lights and resamples the camera image. The system subtracts this image from the background image to generate a difference map, which it then associates with the position and intensity setting of the light. This process is repeated for each of the eight motors (two per light) at ten-degree intervals, leading to a motor-lighting map (which may also be thought of as a sensorimotor contingency table). For each motor position, Elvis creates and stores a lighting map representing the change in lighting location and brightness (see Figure 6). A lighting map is represented as a 320x240 pixel image where each pixel indicates the

intensity level of the light at that spot. Intensities are determined using a 3-tiered system, where red represents 95% to 100% of the maximum detectable brightness level, orange represents 85% to 95%, and yellow represents 75% to 85%. The maximum detectable brightness level is determined by turning on all of Elvis' lights and determining the brightness level of the most intense spot. The system also calculated the weighted center of each spotlight and stores this information for later use by the action planning system.

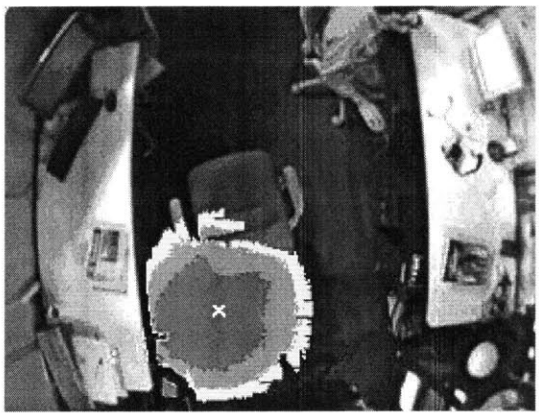


Figure 6. Elvis' spot detection. The white 'X' represents the weighted center of the spotlight.

The procedure takes approximately twenty minutes and needs to be run during initial setup, or whenever a major change is made to the environment (such as rearranging the furniture or moving the light to a new room). Once the training phrase is complete, Elvis is ready to operate in its new environment. Variations in the reflective properties of objects can have some effect on the perceived intensity of a region.

Figure 7 shows a visualization of Elvis' motor-lighting map after training. Each 'X' marks the center of focus for a spotlight position that is stored it its learned sensorimotor map.

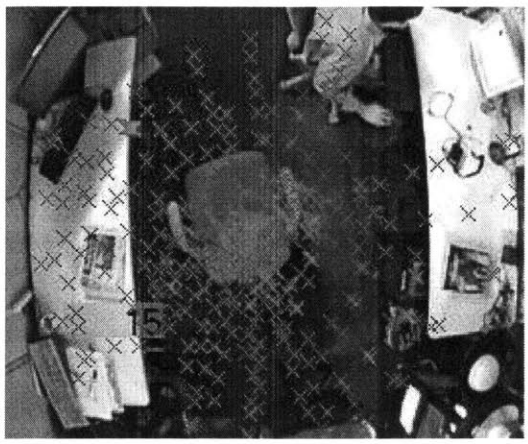


Figure 7. Sensorimotor Map

4. Goal Representation

Elvis views the world as a two dimensional map of lighting intensities. As a result, when Elvis intends on changing its environment, it views this change in terms of a lighting map, where each pixel represents the desired change in state at that point. A change in state can either be absolute or relative. For example, an absolute change might be to set a certain area to half the maximum brightness level. An example of a relative change would be to increase the lighting of an area by 50%. We refer to this map as a *goal*, since Elvis formulates this intermediate representation before utilizing its sensorimotor memory to devise a motor plan.

5. Goal Shifting

5.1 Target Goal State

The target goal state represents the system's beliefs and expectations of the world, which can only be altered by user input. When the system maps a command into a goal, this goal is essentially an instruction on how to change the target goal state.

Since the target goal state represents Elvis' beliefs about the world, it seems reasonable to assume that when Elvis is first turned on, its beliefs about the world are identical to the actual state of the world. The user then changes the target goal state through gesture and speech. Figure 8 gives an example of a target goal state.

Figure 8. Target Goal State. The white region represents an area that was brightened by user input and the gray region represents an area that was darkened.

5.2 Speech Analysis

In this initial implementation, keyword spotting is used to analyze the semantic content of speech. Words are divided into five main categories: actions, areas, conjunctions, intensifiers, and identity.

The author performed several real-world home lighting scenarios in order to create an initial lexicon for the system. Although our lexicon is not comprehensive, it does account for a significant portion of the home lighting domain. In a "Wizard of Oz" user study performed in a home of the future, it was found that, in general, people use two simple elements when controlling lights: an action and a reference. More importantly, it was observed that the two pairs of words, "on" and "off", and "bright" and "dim" account for 81% of lighting-related action words [6].

There are three types of actions in the system. Absolute actions are commands that request an absolute change in lighting, such as "make the room bright". Relative actions are requests for relative changes in lighting such as "dim that area". Finally, label actions are commands that tell the system that the user is naming an area with the subsequent words.

Areas can be either underspecified or predefined. Reference to underspecified areas is signaled by the detection of pronouns such as "this" and "there". These areas must be further specified using the gesture input. Predefined areas are regions that have been previously labeled by the user. To label an area, the user simply gestures over a region and issues a label action. For example, the user circles an area and says "This is my desk". From then on, the system would store the region and the label and whenever the user referred to "my desk", the system would specify the area with the stored region.

Conjunctions are used to split compound sentences into multiple commands or parse out multiple areas in a single command.

Intensifiers are used to modify the degree of effect of an action. For example, a request to make an area "very bright" is translated into a higher level of desired illumination compared to just "bright".

Finally, an identity word is a word used to gain Elvis' attention. Usually this is simply the system's name, such as "Elvis".

When an utterance is completed, it is recognized by the Sphinx speech recognizer and the resulting text is sent to the keyword spotter. It initially searches for conjunctions and divides the sentence accordingly. Each resulting phrase is then parsed for actions, intensifiers and areas. These words are used to fill a command frame, which consists of exactly one action, one or more areas, and an optional intensifier. If there is a conjunction in the sentence but the latter phrase lacks an action word, all areas are added to the prior command frame. For example, the keyword spotter would initially split the sentence, "Darken the table and the chair". It would first create a command frame consisting of the action, "relative-", and the area, "table". When the second phrase, "the chair", is parsed, no action is found and the chair is added to the previous command frame's areas.

If a command frame consists only of an identity word, goal formulation is bypassed and Elvis immediately responds by issuing an acknowledgement. Acknowledgements are requests for attention. Elvis responds by *nodding* its lights (all four lighting elements "nod" by moving in and then out to acknowledge that the robot is ready for multimodal input).

If a command frame lacks an identity word, the command is ignored. This is due to the fact that Elvis requires you to address it when issuing a command. This way, Elvis can always remain attentive and ignore speech that isn't directed at the system.

Otherwise, the action, areas, and intensifier are analyzed in order to create a goal. Areas are defined by gestures, either performed at the time of the utterance or when labeling a region. The only time that a region is not required is when the user refers to the global lighting. Lighting levels are determined by considering the action, the intensifier, and the current light in the region. The system queries a database for an <action, intensifier> pair such as "make much dimmer". If the system locates the exact current lighting level in the database, it is immediately returned a desired level to use for illumination. If the exact level is not available, it interpolates using the two closes lighting levels. This mapping from actions, intensifiers, and current lighting level to expected lighting level was learned through a user interaction study, discussed in further detail in section 5.2.1.

When the frame contains the word "it" in certain linguistic contexts, the system utilizes a simple heuristic to determine whether the area refers to a previous location or the global lighting. For example, if Elvis receives an utterance such as "make it brighter", it will look in its action memory to determine the context of the word "it". If a previous utterance was spoken within its attention span (10 seconds), Elvis will select the previously mentioned location. Otherwise, the system will assume that the user is referring to the entire room. The effectiveness of this feature has not yet been tested, but anecdotal evidence indicates that this is likely a natural behavior.

5.2.1 User Interaction Study: Methodology

Ten subjects were participated in a study inside a mock living room in which Elvis was mounted on the ceiling. Each subject was presented with four lighting

scenarios, each involving varying illumination levels of two distinct elements in the room. The elements included a couch, newspaper, textbook, notebook, floor, ambient light, and two different walls. Figure 9 shows the room from Elvis' perspective.

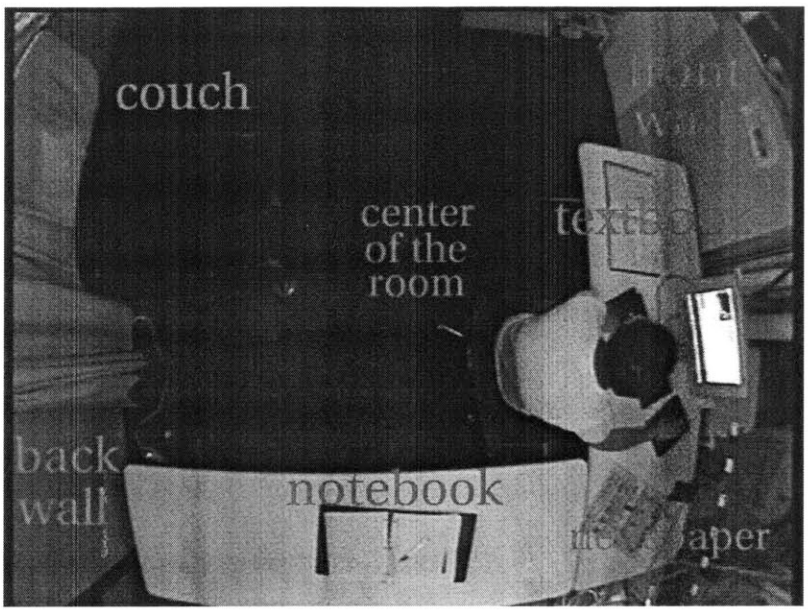


Figure 9. View of room with labeled areas

In each scenario, the user would alternate between making adjustments between each of the two activated elements in the room based on commands issued by a tester who was also present in the room. The interaction went as follows: The tester would announce a random <action, intensifier> pair and an object in the room. For example, "Make the notebook very dim". The subject would then indicate to the tester whether the element needed to be made brighter or dimmer. Using a computer interface, the tester would make adjustments to the lighting level of the element until the subject indicated that he/she was satisfied. The subject would be watching the lighting of the actual object and could not see lighting controls being manipulated by the tester. Additionally, the

subject was free to roam around the room to position him/herself in a desirable location. Each subject was issued and responded to a total of 36 commands. For each command, the system recorded the current lighting level, the <action, intensifier> pair, and the preferred lighting level and added this information to a lookup table.

5.2.2 User Interaction Study: Results

The results of the study were separated into two separate graphs based on their action type: absolute or relative. Figure 10 shows the results for the absolute terms. The system uses the data in this graph by lookup up a before value and an action type, such as 60 and "bright", and then finding the associated after value. This after value is then used as a lighting value for the system.

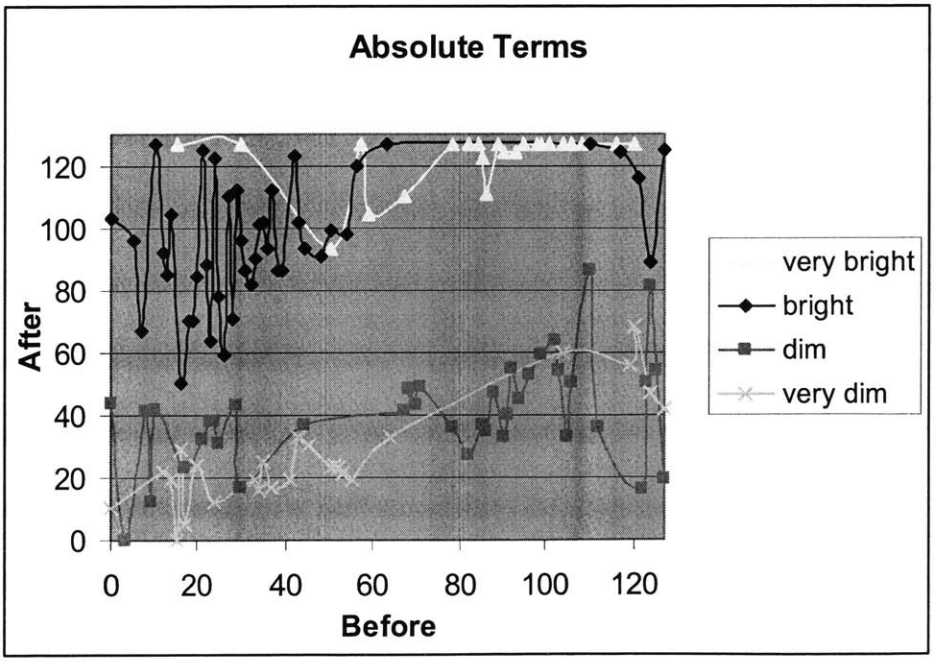


Figure 10. User study results: Absolute Terms

There was a lot of variance in the results; however, generally trends remain in tact. In most cases, the term "very bright" would causes the largest increase in lighting, followed by "bright, "dim", and "very dim", in that order. On the other hand, although there is a large distinction between the two "dim" terms and the two "bright" terms, it seems that the intensifier does not necessarily play a large role. The term "very bright" however often brought the lighting to full power.

Figure 11 shows the results of the relative terms. There was a lot less variance in these terms. Intensifiers had a very clear effect on the actions and, with the exception of a few outliers, the ordering of the results once again accurately reflected the meaning of the term. It should be noted that there were fewer data points for the relative terms.

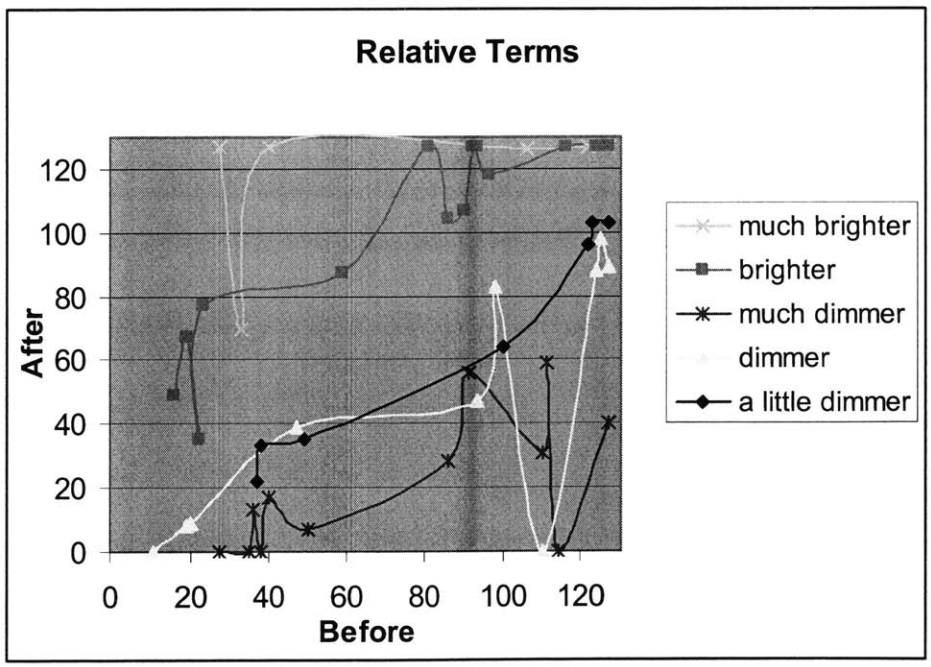


Figure 11. User study results: Relative Terms

5.2.3 User Interaction Study: Discussion

One interesting insight was the fact that it was very common for the user to transform absolute terms to relative terms. For example, the command, "make the room bright" would become "make the room brighter" when a there was already a significant amount of light in the area.

Another noteworthy observation was that most users did not need the presence of an intensifier in order to make drastic changes to the lighting. For example, if the lighting was relatively low and the user was asked to "make it bright", it was not uncommon for the user to request the maximum attainable light level. At first glance, this might be attributed to the fact that the spotlights were in capable of reaching even further extremes, however the same trait was observed while dimming. "Make that dim" often resulted in putting the object in complete darkness.

The variance in the results indicates that there are one or more factors that this study does not take into account. A future study would try to focus on two areas which were simplified for this study: object reflectance levels and context. By choosing a large range of objects, the effect of variations in object reflectance can be reduced through averaging. However, in some cases, differences were somewhat drastic. A spotlight shining on a black couch, for example, had much less influence on its lighting than the bleached white pages of a textbook. As a result, it was not uncommon for a user to max out the brightness levels of the couch while only requesting half-brightness when addressing the pages of the textbook. This effect could be accounted for if Elvis learned the range of lighting values possible for a given region and recording lighting changes as percentages of that range.

Additionally, context was not considered at all. The system doesn't consider the fact that the user requested light on a textbook so that the pages could be legible versus light on a wall to set the mood. This distinction might play a vital role in how adjustments are made and might be worth looking in future studies.

5.3 Gesture Analysis

The description of a lighting environment primarily deals with transitive and intransitive deictic gestures [3] such as pointing and waving. The situation is not that much different from that of a weatherperson [4]. Instead of gesturing in front of a screen while facing a camera, the user is gesturing above a physical location while the camera observes from above. As a result, an effective system would likely need to be able to recognize the same primitive gestures used by Kettebekov and Sharma's iMap system [5]. These primitives are *pointing, circle,* and *contour* gestures. For this implementation, we make two modifications to these primitives. Firstly, the current version does not yet recognize motion actions such as "move the light to the right." As a result, the *contour* gesture isn't applicable. Second, because the recognition system only recognizes two-dimensional regions, a general pointing gesture could lead to very ambiguous situations for the following reason. The pointing gesture is most effective when one visualizes a three-dimensional vector protruding from the end of the hand and intersecting the desired location. With such limited information, the system would not know at what point this vector intersects an object. It would, for example, be unable to tell the difference between a user pointing at a wall or the table in front of it. As a result, the system recognizes a very simplified version of pointing in which it interprets the user to be indicating an area directly below his or her moving hand. This simplifies the gesturing system dramatically

by handling both pointing and circling in the same manner: detect the region that is contained within the perceived motion.

Gesturing is performed using a small red LED wand. A button must be held down on the wand while the user's hand is in motion. Although it is not as natural as using untethered gesture, the technique is quickly learned and becomes quite intuitive. One added benefit of the method is that recognition is robust in a variety of lighting conditions: a crucial requirement for a robot that's sole purpose is adjusting the lighting properties of its environment.

Bright red spots are detected every 60 ms. and stored in a vector of points. When the gesture is complete (no red light is detected for at least a half a second), the convex hull of the points is calculated using the QuickHull algorithm [10]. The points are then sorted and a polygonal region is formed. This area is then stored in a buffer. If a command frame requires an area, it looks in the buffer to find the most recent gesture. Figure 12 shows an image from Elvis' point of view of a user gesturing and the resulting convex hull.

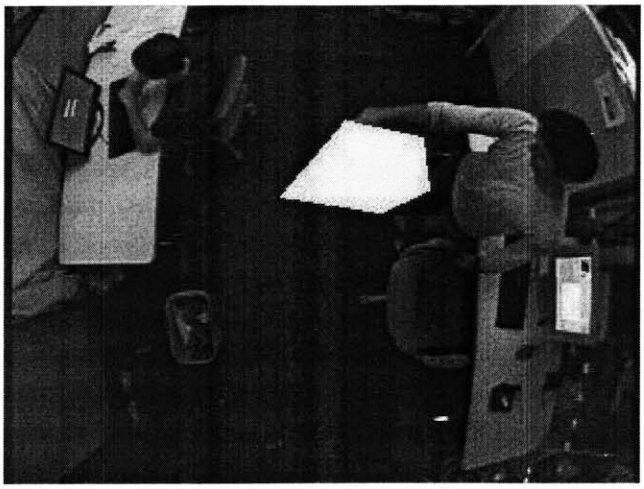


Figure 12. Gesture recognition and resulting convex polygonal region

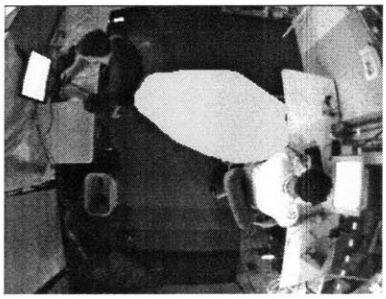
6. Goal Maintenance

Elvis is constantly monitoring its current lighting conditions and makes adjustments to the environment when these conditions differ greatly from its expectations. Once a change is detected, the system adjusts its lighting environment so that it is more closely aligned with its beliefs about the world. It is not uncommon for Elvis to be incapable of eliminating this gap between perceptions and expectations; however the system minimizes the difference to the best of its ability. In order to determine how to successfully manipulate the world, the system utilizes another property that is commonly found in first-order cybernetic systems: simulation. By accessing its sensorimotor contingency table and its memory of how its own lights are influencing the world, the system attempts to simulate a lighting scene that most closely matches its target goal state (i.e. its expectations of the world). This will be described in great length in the following sections.

6.1 Difference Maps

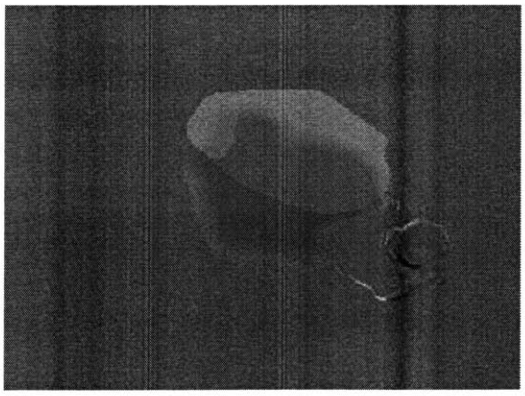
Up to now, we have discussed the fact that Elvis is always monitoring the current lighting environment and comparing it to its target goal state. Now, we will focus on exactly how this process takes place. In simple terms, a difference map is the world state subtracted from the target goal state. Areas that are brighter than expected are indicated by negative pixel values in the difference map (indicating that light should be reduced), while regions that are darker than expected (that need additional light) are indicated by positive pixel values. Figure 13 shows an example of this initial difference map obtained

after a user gestured for light in the middle of the room and Elvis responded. The background color (dark gray) represents regions in which the target goal state and the world state match up. Regions that are darker than the background indicate areas where there is too much light and brighter regions indicate areas where there needs to be more light. In this example, Elvis covers the region very well (it is using a combination of three spotlights), however it shines its lights too brightly.



(a) The target goal state

(b) The world state



(c) The uncorrected difference map

Figure 13

As mentioned earlier, the system chooses a light and motor plan by utilizing the difference map. However, if the system were to use the difference map in Figure 11, its next plan of action would be to turn off the lights covering the area because the difference

map indicates these regions are too bright. To account for this, Elvis must subtract regions of light that it knows are caused by its own spotlights. To do this, the system stores the difference between images taken directly before and after an action is taken place. The resulting differences are then added back into difference map. The corrected difference map can be seen in Figure 14. Notice that now Elvis will attempt to formulate a plan that once again shines light on the marked region. If no other changes have been made to the environment or target goal state, the plan would presumably be the same and no changes will be made to the light configuration.

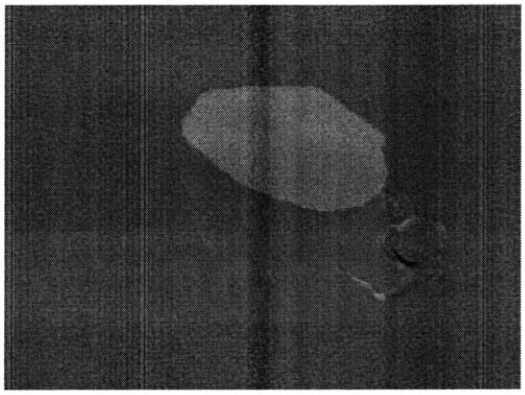


Figure 14. A corrected difference map.

The system computes the difference map several times a second. Since it is unable to make regions darker, it only is concerned with regions with positive values (light gray areas). The next few sections describe how the system converts this difference map to motor and lighting configurations.

6.2 Connected Components

The first step of analysis is to determine regions where lighting is most needed. The system uses a connected component algorithm [7] based on 8-connectivity to segment spatially connected regions. As described by Park, J.M. et al [8]., (we do not use

the divide-and-conquer method) an initial labeling is done by scanning the image and marking neighboring pixels that are above a certain threshold with the same labels. Label equivalences are then resolved by first calculating reflectivity (if label A = label B, label B = label A) and then computing the transitive closure using the Floyd-Warshall algorithm [9].

6.3 Choosing the N Best Points

The next step is to find the best points to initially place the lights. To do this, the system uses a greedy algorithm to select the optimal quantity and location of points. During the sensorimotor learning phase, the system average size of a blob that spotlight makes. It uses this number to determine how many points to create. Say for example, the largest blob consists of 12,000 pixels. It would add a point to this blob and subtract the average blob size (5388 pixels) from the chosen blob. This blob, with a modified size, is then added to the list of connected components and re-sorted. This process is repeated until no blob is at least half the size of the average spotlight or the system runs out of lights. The points are initially placed in the weighted center of each blob.

6.4 Devising a Motor Plan

Although the system has chosen the best points, it has yet to assign actual spot lights to them. To do this, the system, performs a depth-first search until all points are assigned valid spotlight positions. Initially, the system checks the sensorimotor contingency table to see if a spotlight's center is located in the exact location of the point. If not, the system continues to checks the neighboring pixels. It continues to increase its

search radius until valid spotlight positions are found. If the radius becomes too large, the search fails and the system moves checks the remaining spotlights. In order for a light to be selected, it must meet two conditions:

(1) the light is not currently selected in the lighting plan

(2) the light will not interfere with its neighboring lights

The resulting preliminary lighting plan is fairly good, however much improvement can be made. Firstly, the system only compares the calculated centers of spots and regions. More importantly, regions which require multiple lights place all points at the exact same location in the blob. The next section described how the system optimizes the lighting plan to account for these deficiencies.

6.5 Optimizing the Results

Up to this point, the only use of the lighting maps in the sensorimotor contingency table was for determining the center of spots. This was very useful in coming up with a preliminary plan, however the system must now take full advantage of its sensorimotor contingency table to come up with an optimal plan. To do this, the system combines the all of the sensorimotor maps in the current lighting plan. The system uses a simple bounded, additive model to combine the 3-tiered sensorimotor maps (described in Section 3). Since the brightest tier was supposed to be close to the maximum brightest spot that Elvis can achieve, all values are capped at that level. The resulting map is then given a normalized score based on how closely it matches the difference map.

The system repeats this process by moving each spotlight configuration in different directions. If the score improves, the new location is incorporated into the

configuration. Since the system does not allow regions to be brighter than the brightest tier, overlapping bright spots from multiple spot lights are less favorable than separated spots. As a result, the duplicate points found in the center of large blobs are shifted to more suitable locations.

6.6 Discussion

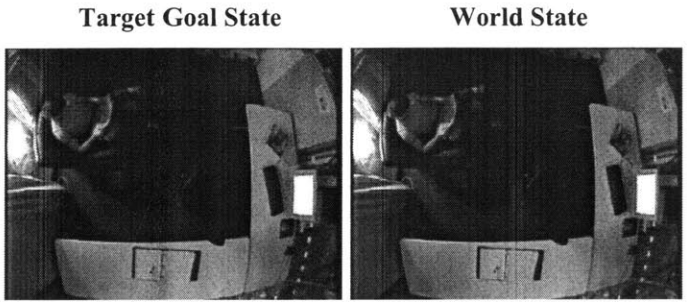
Speed was a critical issue when designing the goal maintenance system. The Elvis monitors and updates its state several times a second and, consequently, it must formulate a viable plan within each of those periods. Unfortunately, at the current time, there is not enough time for a state-of-the-art computer system to perform a search algorithm that involves comparing over 700 pixel maps to one another.

One major problem found in the current version of Elvis' goal maintenance system is that it does not utilize any context from the scene. One of the most obvious issues is that Elvis doesn't keep track of people. When a person is wearing light colored shirt and moves in a scene, the original location is viewed by the system as having darkened dramatically while the new location has increased in brightness. As a result, a light is turned on to try to compensate. This behavior is undesirable but could be corrected if the system was aware of people. If Elvis could track people, it would know not to monitor changes in brightness that are due to motion.

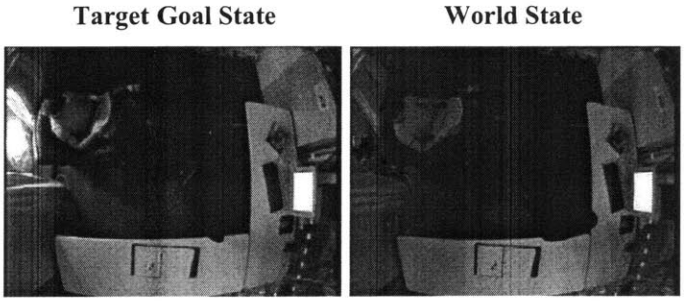
7. Sample Interaction

This section illustrates a typical interaction with the system by the author, Josh.

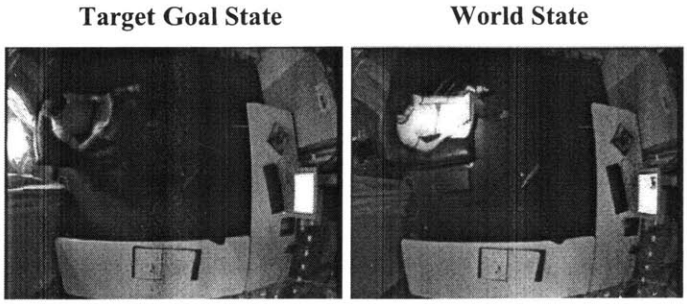
(1) Josh enters a room, sits on the couch, and starts reading a newspaper. Unfortunately, the sun was setting and Josh's natural light source was slowly disappearing. At first, it was bearable, but after some time, Elvis added light to Josh's reading area without him having to issue any commands. The light slowly increases in brightness as the sun finishes its course. Figure 15 shows how Elvis maintains its target goal state.



(a) Homeostasis



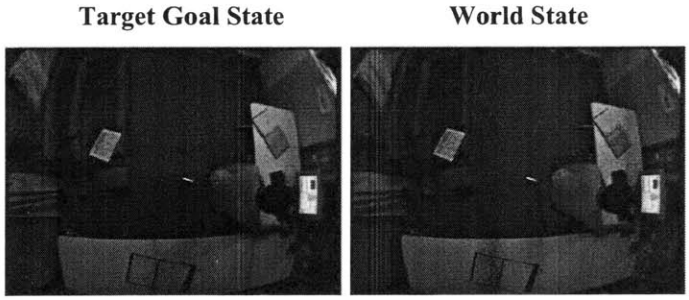
(b) Homeostasis Perturbed



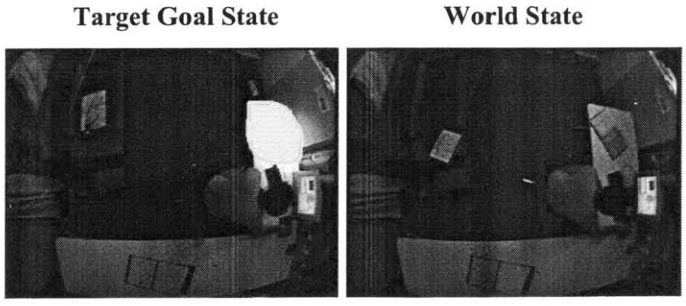
(a) Homeostasis Achieved

Figure 15. Example of changes to the environment perturbing homeostasis

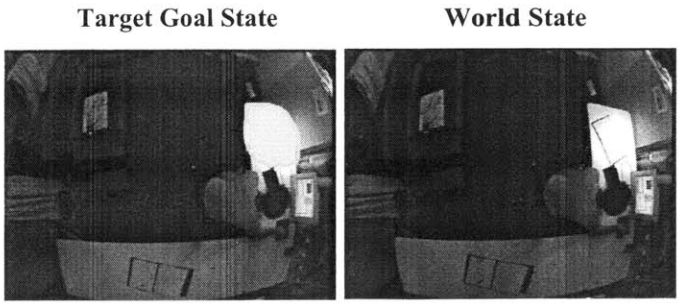
(2) Next, Josh decides to sit at his desk and requests some light so he can read his textbook. Josh labeled his desk using speech and gesture when he originally set up the system. He firsts shouts out Elvis' name. Elvis immediately twitches, letting Josh now that he is paying attention. Josh says, "Make my desk very bright.." At this point, Josh's target goal state has been altered. A gesture region was obtained from the system's object store and combined with the speech to form a goal, which is then integrated into the target goal state. Noticing a large discrepancy in Elvis' difference map, it immediately swings a light around and shines on the desk. Figure 16 shows this process.



(a) Homeostasis



(b) Homeostasis Perturbed



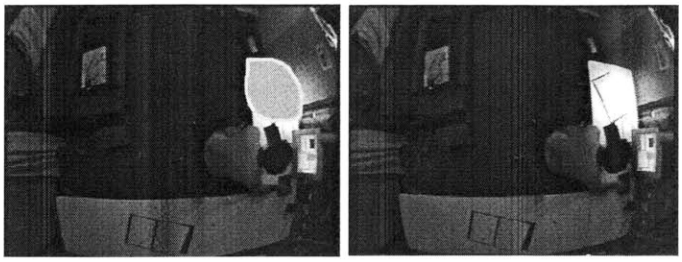
(c) Homeostasis Achieved

Figure 16. Example of using speech and object labeling to perturb homeostasis

(3) Unfortunately, Elvis made the desk too bright. Josh simply says, "Make it dimmer". Since Elvis was still paying attention, there was no need to address it by name. Additionally, because it was within its attention span, Elvis used the pronoun "it" as a placeholder for the previously mentioned area. Once again, homeostasis is perturbed. The area where the notebook is brighter in the world state compared to the target goal state. Elvis comes up with an entirely new plan, however, not surprisingly, the light remains in the same but is dimmed. Figure 17 shows this process.

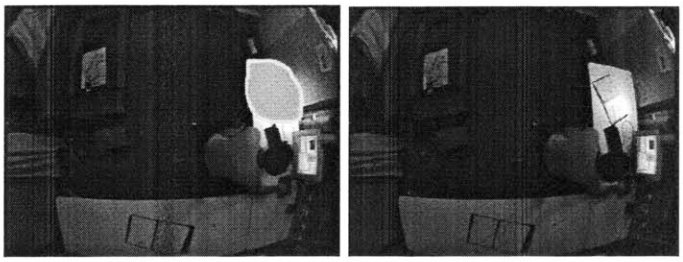
(a) Homeostasis

Target Goal State World State



(b) Homeostasis Perturbed

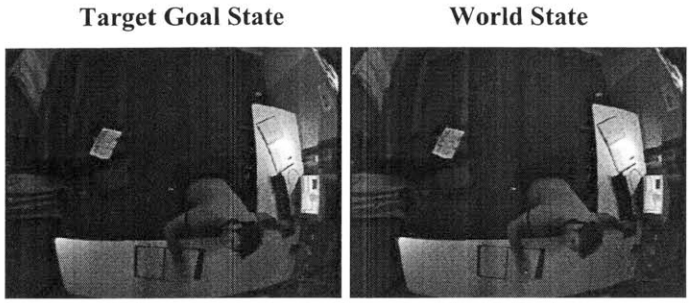
Target Goal State World State



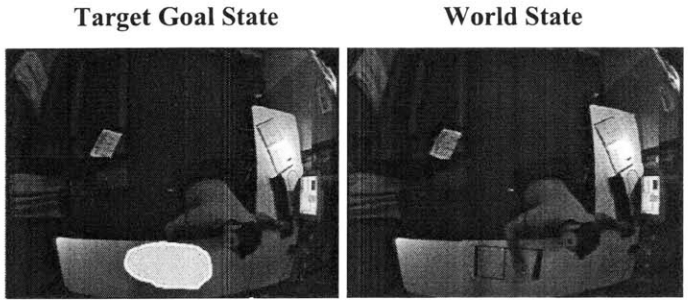
(c) Homeostasis Achieved

Figure 17. Example of use of pronouns and the Elvis' attention system

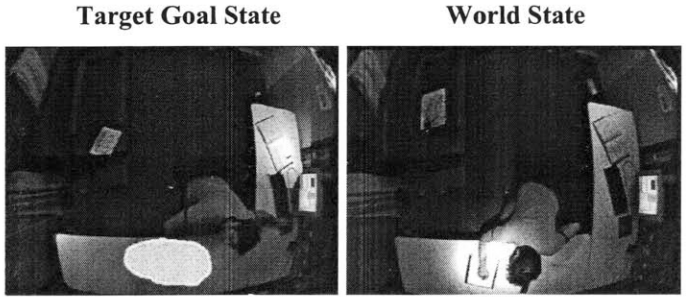
(4) Finally, Josh decides to write in his notebook. Unfortunately, he never labeled the area where the notebook was sitting. He uses his gesture want to and waves his hand above the area. While doing this, he says, "Elvis, shine some light over here." Elvis correctly identifies the region and updates its target goal state. Noticing that homeostasis is once again perturbed, Elvis shines light on the notebook. See Figure 18.



(a) Homeostasis



(b) Homeostasis Perturbed



(c) Homeostasis Achieved

Figure 18. Example of speech and gesturing to perturb homeostasis

8. Conclusion

We have presented a working robotic lighting system that translates speech and gesture commands into lighting changes. The system uses a set of four actuated lighting elements and a fifth fixed ambient light to set lighting scenes according to the user's

requests. The basic control architecture of the system is based on the cybernetic notion of homeostasis. Speech and gestures perturb the robot's desired goal state, whereas environmental changes perturb the robot's perceived world state. Either type of perturbation causes the robot to take appropriate actions to regain homeostasis.

We believe that a homeostasis control framework is a promising approach for the design of a variety of situated, interactive systems in which a layered approach to interface design may be used to create natural multimodal interfaces.

REFERENCES

[1] P. Lamere, P. Kwok, et al. *Design of the CMU Sphinx-4 Decoder.* Eurospeech, September 2003.

[2] M. Kuperstein. *Neural model of adaptive hand-eye coordination for single postures.* Science, 239. 1308-1311, 1988.

[3] D. McNeill. *Hand and Mind.* The University of Chicago Press, Chicago, 1992.

[4] R. Sharma, J. Cai, S. Chakravarthy, I. Poddar and Y. Sethi. *Exploiting Speech/Gesture Cooccurrence for Improving Continuous Gesture Recognition in Weather Narration.* In Proc. International Conference on Face and Gesture Recognition, Grenoble, France, 2000.

[5] S. Kettebekov and R. Sharma. *Understanding Gestures in Multimodal Human-Computer Interaction.* International Journal on Artificial Intelligence Tools, vol. 9, no. 2, pp. 205-224, June 2000.

[6] B. Brumitt and J. Cadiz. *"Let There* Be *Light!"Comparing Interfaces for Homes of the Future.* Technical Report MSR-TR-2000-92. September 2000.

[7] Rosenfeld, A., Pfaltz, J.L., "Sequential Operations in digital Processing," JACM, 13, 471-494, 1966.

[8] Jung_Me Park, Carl G. Looney, Hui_Chuan Chen, "Fast Connected Component Labeling Algorithm Using A Divide and Conquer Technique." Technical report, 2000. (http://cs.ua.edu/TechnicalReports/TR-2000-04.pdf).

[9] R. W. Floyd, "Algorithm 97: Shortest path," C.ACM, 5, 6, pp. 345, 1963.

[10] C. Barber, D. Dobkin, H. Huhdanpaa, "The Quickhull Algorithm for Convex Hull", Geometry Center Technical Report GCG53, Univ. of Minnesota, MN, 1993.

Appendix A. Lexicon

Relative+
brighter
lighter
more bright
less dim
less dark
shine light on
shine more light on
shine some light on
can you shine some light on
can you shine light on
how about some light on
give me some light on
brighten up
lighten up
lighten
brighten

Relative-
less light
less bright
darker
dimmer
more dim
lower the light on
darken
dim

Absolute+
shine
bright
very bright
very light
light up

Absolute0
dark
remove the light from
remove light from
remove light on
remove the light on
shine no light on
get rid of the light on

Absolute-
very dim

Conjunctions
but
and also
and

Intensifier+
a lot
a ton
a great amount
a huge amount
very
much
extremely

Intensifier-
a small amount
a little bit
a tiny bit
a bit

Areas
it
over (t)here
that
this
(this | that) way
(t)here
room
ambient
(this | that) area
(this | that) region
(this | that) part
them
those
these

Appendix B. Implementation Details

B.1 System Architecture

The system was written entirely in Java using Sun's API v1.4.2. As seen in Figure 19, the systems centers on the LightChecker module. This module constantly generates difference maps, which is then used by the ActionPlanner module to predict a motor plan. The ActionCoordinator module controls the lights and motors and actually implements the motor plan to perform an action.

On the other side of the diagram, we see that a goal is formed from a command frame and either a gesture or an object map.

The modular nature of the system makes it very easy and intuitive to make large changes to the system. For example, originally the lights were controlled using X10 lighting control modules through home power lines. However, this technique proved to be very slow and was replaced with a MIDI control module. In practice, this significant change only required minor changes to be made to the Light class.

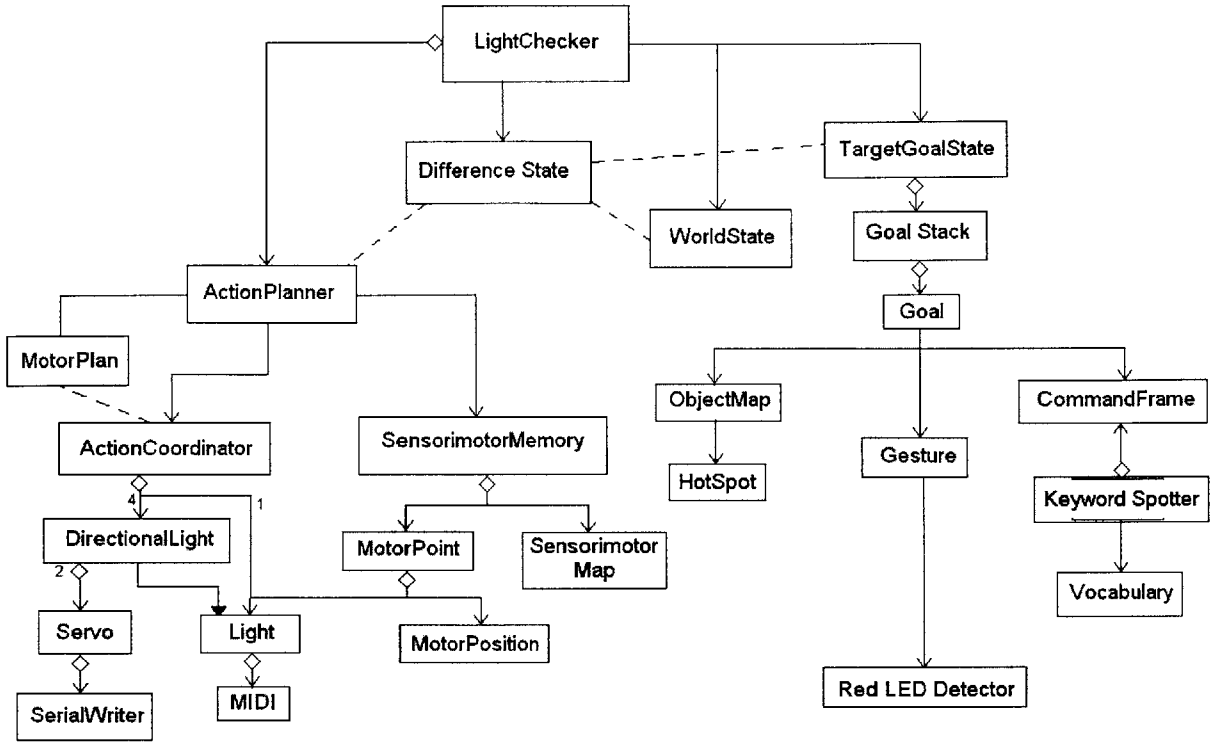


Figure 19. Object Model

B.2 Multi-system and Multi-threaded

Elvis' code base and the Sphinx recognizer are run on separate machines due to memory and CPU constraints. The two modules talk to one another via the PVM packet protocol.

The system has multiple threads running at all times. A PVM packet listener constantly awaits text from the speech recognizer. Another thread is monitoring the world state to locate bright red spots from the gesture wand and create new areas. One thread constantly updates the video input. Additionally, another module constantly generates difference maps, formulates plans, and performs actions.

B.3 Vocabulary

A design decision was made to store all vocabulary in a text file using a markup language consisting of a few different tag types. They include the following: <name>, <action:relative+>, <action:relative->, <action:absolute+>, <action:absolute->, <action:absolute0>, <action:label>, <conjunction>, <intensifier+>, <intensifier->, and <objects-and-areas>. Simply adding a new word or phrase to one of these categories would enable it to be correctly categorized by the keyword spotting system. A utility was also written to convert this file format to a format readable by the grammar generator for the speech recognition system.